

Sprint 3 - Agility Design Document

Table of Contents

1. EXECUTIVE SUMMARY	3
1.1 PROJECT OVERVIEW	3
1.2 PURPOSE AND SCOPE OF THIS SPECIFICATION	3
2. PRODUCT/SERVICE DESCRIPTION	3
2.1 PRODUCT CONTEXT	3
2.2 USER CHARACTERISTICS	3
2.3 ASSUMPTIONS	3
2.4 CONSTRAINTS	3
2.5 DEPENDENCIES	4
3. REQUIREMENTS	4
3.1 FUNCTIONAL REQUIREMENTS	5
3.2 SECURITY	5
3.2.1 <i>Protection</i>	5
3.2.2 <i>Authorization and Authentication</i>	6
3.3 PORTABILITY	6
4. REQUIREMENTS CONFIRMATION/STAKEHOLDER SIGN-OFF	6
5. SYSTEM DESIGN	6
5.1 ALGORITHM	6
5.2 SYSTEM FLOW	6
5.3 SOFTWARE	6
5.4 HARDWARE	6
5.5 TEST PLAN	7
5.6 TASK LIST/GANTT CHART	7
5.7 STAFFING PLAN	7

1. Executive Summary

1.1 Project Overview

A robot is set to navigate an obstacle course, commencing within a square. Along its path, the robot will confront three objects necessitating avoidance. Subsequently, it will traverse a ramp before concluding the course by attempting to topple as many pins as it can. Points will be awarded for successfully navigating each obstacle, avoiding obstacles, and knocking down pins.

This project is intended for students who want to show off the abilities of their robots. It's an entertaining method to evaluate robot capabilities and draw in tech and automation enthusiasts.

<https://monmouth.desire2learn.com/d2l/le/content/316749/viewContent/3838883/View>

1.2 Purpose and Scope of this Specification

Specification Description:

Purpose:

This specification document outlines the requirements and constraints for the implementation of the third phase of Project XYZ, specifically focusing on modifications to the Classification Processing and Labor Relations Processing components. The primary purpose of this document is to provide clear guidance to the project team and stakeholders regarding what is expected in this project phase.

Intended Audience:

The intended audience for this specification includes project stakeholders, project managers, developers, quality assurance teams, and anyone involved in the planning and execution of Project XYZ. It serves as a reference for those responsible for implementing and monitoring the project.

In Scope:

This document addresses requirements related to Phase 2 of Project XYZ, which includes:

Modification of Classification Processing: This phase involves making changes to the Classification Processing component to align with the legislative mandate ABC. This may include updating algorithms, data structures, or interfaces to ensure compliance with the new requirements.

Modification of Labor Relations Processing: Similarly, this phase requires modifications to the Labor Relations Processing component to meet legislative mandate ABC. This could involve adjustments to data handling, workflow processes, or reporting mechanisms.

Out of Scope:

The following items are explicitly out of scope for Phase 2 of Project XYZ:

In summary, this document serves to provide a clear understanding of what the current project phase (Phase 2) will and will not cover, ensuring that project stakeholders are aware of the scope and limitations of this specific phase, with a clear plan for addressing other legislative mandates in subsequent project phases.

2. Product/Service Description

Key Factors Shaping Product Requirements

To comprehend the specific requirements, it's crucial to consider these overarching factors:

Legal Obligations: The product must meet legal mandates, avoiding potential penalties.

Project Scope: Understand what this project phase covers (Phase 2) and what's for later (e.g., Phase 3).

Sprint 3 - Agility Design Document

Technical Limits: Consider the technology's capabilities and constraints, like compatibility and performance.

Budget Limits: Financial resources impact the scope and scale of the product.

Project Timeline: The schedule affects prioritization and sequencing of requirements.

Stakeholder Needs: Meet user and regulatory expectations to ensure success.

Risk Management: Requirements may address identified risks for stability and reliability.

Industry Standards: Adherence to industry norms ensures compatibility and quality.

User Experience: Usability is critical for user satisfaction.

Scalability: Ensure the product can adapt to future changes.

Environmental Impact: Environmental considerations may affect requirements due to regulations or ethical concerns.

These factors underpin specific requirements and guide product development.

2.1 Product Context

The product typically connects with other systems and components, forming a larger system. It might link to databases, hardware, external services, and user interfaces. These connections are essential for the product's functionality and data exchange, ensuring it works seamlessly in a larger ecosystem. Diagrams are often used to visually represent these connections.

2.2 User Characteristics

Student Profile:

Type: Student

Experience: Varies, from beginners to more experienced.

Technical Skills: Basic to intermediate.

Characteristics: Value user-friendliness, efficiency.

Faculty Profile:

Type: Faculty

Experience: Experts in their fields.

Technical Skills: Varied, from basic to advanced.

Characteristics: Prioritize functionality and reliability.

Staff Profile:

Type: Staff

Experience: Experienced in administrative roles.

Technical Skills: Varies, basic to role-specific.

Characteristics: Value efficiency and accuracy.

Other Users (e.g., Researchers, Alumni, Visitors):

Type: Specify user group.

Experience: Varies by group.

Technical Skills: Match specific needs.

Characteristics: Vary based on user group's purpose and requirements.

2.3 Assumptions

Assumptions Impacting Requirements:

Sprint 3 - Agility Design Document

Equipment Availability: Assumes users have necessary hardware and software. If not, the product must be flexible.

Internet Connection: Assumes users have internet access. If not, product should work offline or provide alternatives.

User Expertise: Assumes users have some tech skills. Product must be user-friendly and offer guidance for less tech-savvy users.

Operating System Compatibility: Assumes compatibility with common operating systems. The product may need to support more platforms.

Security Assumptions: Assumes basic user security practices. The product may need stronger security measures.

Data Privacy Compliance: Assumes adherence to privacy regulations. Product may need to adapt to different laws.

Third-Party Services: Assumes reliance on certain services. Contingency plans are needed for service disruptions.

Resource Constraints: Assumes available server and storage resources. Adaptations may be required for resource limitations.

Language and Localization: Assumes primary language. The product should support multiple languages and locales.

User Behavior: Assumes user patterns. Product should adjust engagement strategies for different behaviors.

2.4 Constraints

Parallel Operation with an Old System:

The product must operate in parallel with an existing legacy system during a transition period. This constraint requires the new product to seamlessly integrate with the old system, ensuring data consistency and user continuity during the migration phase.

Audit Functions:

The product must incorporate audit functions, including audit trails and log files, to track user actions, system activities, and data modifications. These functions are crucial for compliance, security, and accountability, and must be integrated into the design.

Access, Management, and Security:

Access control and security are paramount. The product design must adhere to strict access control policies, ensuring that users can only access authorized features and data. It should also implement robust data encryption and protection measures to safeguard sensitive information.

Criticality of the Application:

The application's criticality is high, and any system downtime or malfunction can have severe consequences. The design should prioritize fault tolerance, redundancy, and disaster recovery mechanisms to minimize disruptions.

System Resource Constraints:

The system operates under specific resource constraints, including limited disk space and hardware limitations. The design must optimize resource usage and scalability to ensure efficient performance within these constraints.

Other Design Constraints:

The product design must adhere to specific standards and guidelines, including programming languages, frameworks, and design principles specified by the organization or industry. Compliance with these standards is non-negotiable.

Sprint 3 - Agility Design Document

These design constraints are essential factors that shape the product's design and functionality. They influence decisions regarding compatibility with legacy systems, security measures, data management, and adherence to industry standards, ultimately ensuring that the product meets its intended purpose and complies with necessary requirements and regulations.

2.5 Dependencies

Data Integration Dependency:

The new product will require daily data downloads from an external source (System X) to update information. Requirements must specify the data format, data transfer methods, and data quality standards.

Module Dependency:

Module X needs to be completed before the development of this module can begin. Requirements should detail the functionalities and interfaces that rely on Module X, ensuring a smooth integration.

Third-Party Service Integration:

The product depends on integrating with a third-party service (Service Y) for specific features. Requirements should define the integration protocols, data exchange formats, and access credentials needed for this integration.

Operating System Dependency:

The product must run on a specific operating system (e.g., Windows or Linux). Requirements should address compatibility and specify the required features or libraries associated with the chosen operating system.

Regulatory Compliance:

Compliance with certain regulatory standards or industry-specific requirements is necessary. The product's requirements should align with these regulations, detailing how the product will meet these standards.

Hardware Dependency:

The product's performance may rely on specific hardware components, such as sensors or GPUs. Requirements should outline the hardware specifications and configurations needed to support the product's functionality.

Security and Authentication Services:

The product requires integration with an organization's security and authentication services. Requirements must specify the authentication protocols, access control mechanisms, and encryption standards to be used.

API Availability:

Certain features of the product depend on the availability and functionality of external APIs. Requirements should define the API endpoints, authentication methods, and data exchange protocols for integration.

Data Availability and Quality:

The product depends on the availability and quality of certain data sources. Requirements should address data source specifications, data validation processes, and backup strategies in case of data unavailability.

Resource Constraints:

Resource constraints, such as limits on server resources or network bandwidth, affect the product's performance. Requirements should specify resource usage and optimization guidelines to meet these constraints.

These dependencies are critical considerations when defining product requirements, as they influence the product's design, functionality, and performance.

3. Requirements

Software Development Project - CS104 Robotics Triathlon Requirements

Sprint 1 - Endurance Requirements (Priority 1)

1.1 The robot must successfully circumnavigate the periphery of HH208.

- Input: Clear path provided from each outside wall.

Sprint 3 - Agility Design Document

- Output: The robot starts from the yellow square with blue tape, displays a green light, and speaks "ready set go." It stops with a red light and speaks "I'm done and I need water."
- Function: The robot should navigate to each yellow floor tile and turn right at the center of each tile. It must return to its starting location.
- Location: HH208 periphery.
- User: Judges and participants.

- 1.2 Points will be deducted if the robot fails to light and speak at the start and finish.
- 1.3 Points will be deducted if the robot collides with any objects.
- 1.4 Points will be deducted if the robot does not finish in the square where it started.

Sprint 2 - Accuracy Requirements (Priority 1)

- 2.1 The robot must successfully run the figure-eight course five times.
 - Input: A laid-out path on the floor.
 - Output: The robot starts and finishes in the provided square, speaks "I am the winner," and flashes multicolored lights for 5 seconds.
 - Function: The robot must stay within the provided path.
 - Location: Figure-eight course area.
 - User: Judges and participants.

- 2.2 Points will be deducted if the robot strays from the path.
- 2.3 Points will be deducted if the robot does not complete five runs.
- 2.4 Points will be deducted if the robot does not finish in the same place it started.

Sprint 3 - Agility Requirements (Priority 1)

- 3.1 The robot must navigate an obstacle course.
 - Input: A predefined obstacle course layout.
 - Output: The robot successfully avoids three objects, goes over a ramp, and knocks over as many pins as possible.
 - Function: The robot should demonstrate agility by avoiding obstacles and toppling pins.
 - Location: Obstacle course area.
 - User: Judges and participants.

- 3.2 Points will be added for each obstacle the robot completes.
- 3.3 Points will be added for each obstacle avoided.
- 3.4 Points will be added for each pin the robot topples.

General Requirements (Applicable to all sprints)

- 4.1 Use a Github repository for all project elements.
- 4.2 The System Design Document must contain all sections: Executive Summary, Product Description, Requirements, Requirements Confirmation, System Design (with Algorithm, Flowchart, Hardware/Software, Test Plan, Test List/Gantt Chart, Staffing Plan), Code, and Presentation on Github.

System Constraints

- 5.1 The robot must operate using the specified hardware components and programming languages.
- 5.2 The robot's design must ensure stability and reliability.
- 5.3 Compliance with safety regulations is mandatory.

Dependencies

- 6.1 The project is dependent on the availability of required hardware components and materials.
- 6.2 The completion of each sprint is dependent on the prior sprint's successful completion.
- 6.3 The Github repository must be accessible and up-to-date for collaborative work.

Traceability

Sprint 3 - Agility Design Document

- 7.1 Each requirement should be traceable to its source document or specific project objective.
- 7.2 Changes to requirements should be documented and tracked through a formal change process.

These requirements are structured to provide a clear overview of the specific functionalities, inputs, outputs, and priorities for the CS104 Robotics Triathlon project. The traceability and documentation elements ensure that changes are managed and the project is well-organized.

3.1 Functional Requirements

Req#	Requirement	Comments	Priority	Date Rvwd	SME Reviewed / Approved
FR-001	The robot must successfully navigate the square course	Verify that the robot completes the square without error.	High	12/4/23	12/4/23
FR-002	The robot must stay within the predefined path laid out on the floor during each run.	Observe and confirm that the robot remains within the specified path throughout each run.	High	12/4/23	12/4/23
FR-003	The robot must start and finish within the designated square on the floor.	Ensure that the robot initiates and concludes each run inside the provided square.	High	12/4/23	12/4/23
FR-004	The robot must go over the ramp	It has to traverse over the ramp and maintain on the path of the course.	Medium	12/4/23	12/4/23
FR-005	Then the robot will encounter 3 objects which it must avoid	The robot has to change direction/ swerve away from the the objects.	High	12/4/23	12/4/23
FR-006	Points will be deducted if the robot strays from the predefined path.	Simulate scenarios where the robot deviates from the path, and verify that the appropriate points are deducted.	medium	12/4/23	12/4/23
FR-007	Points will be deducted if the robot does not finish in the same place it started.	Confirm that points are deducted when the robot does not conclude a run in the initial starting square.	low	12/4/23	12/4/23
FR-007	The robot will knock over as many pins as possible	The robot must successfully collide and topple over pins.	High	12/4/23	12/4/23

3.2 Security

3.2.1 Protection

To protect the CS104 Robotics Triathlon system from unauthorized access or harm, consider these simplified security measures:

1. User Verification: Ensure only authorized users can control the robot or modify the system.

Sprint 3 - Agility Design Document

2. Data Encryption: Secure data when it's sent or stored using codes.
3. Record Activity: Keep a log of all actions and data for reference.
4. Access Control: Limit what different users can do within the system.
5. Check Data Integrity: Make sure data remains unchanged during transfer.
6. Network Protection: Guard the system from outside threats with firewalls and filters.
7. Safe Communication: Use secure communication methods.
8. Regular Security Checks: Regularly look for vulnerabilities.
9. Intrusion Detection: Detect suspicious activities and report them.
10. Physical Security: Protect the robot and system from physical tampering.
11. User Training: Educate users about security practices.
12. Safe Updates: Securely deliver and install updates.
13. Response Plan: Have a plan in place to handle security incidents.

These measures safeguard the system from both accidental and malicious threats, ensuring data and system integrity.

3.2.2 Authorization and Authentication

In the context of the CS104 Robotics Triathlon project, authentication and authorization can be crucial for ensuring the security and controlled access to the robot's software, systems, and data. Here's how they relate to the project:

Authentication in the Project:

Robot Access: Ensure that only authorized team members can access and control the robot. Authentication mechanisms may include password-based access to the robot's control software.

GitHub Repository: Authenticate team members who have access to the project's GitHub repository, ensuring that only authorized individuals can make code changes.

Test Robot Access: Authenticate technicians or team members who are responsible for configuring and testing the physical robot.

Authorization in the Project:

Access Control for Code Development: Use authorization to control who can make changes to the project's code. For instance, project leads may have full access, while other team members may have read-only access.

Test Plan Execution: Authorize specific team members to execute and document test cases. Not all team members may have this authorization.

System Design Document Review: Authorization can determine who has the right to review and approve sections of the System Design Document, ensuring that only designated individuals can make decisions about system design.

GitHub Repository Access: Control access to the GitHub repository based on the roles and responsibilities of team members. Some may only need read access, while others require write access to commit code.

Sprint 3 - Agility Design Document

Robot Control: Authorize specific team members to control the robot during the triathlon events, ensuring that only authorized operators can initiate robot actions.

In this project, authentication and authorization help maintain security and control over various aspects of the project, such as code development, testing, and robot operation. They ensure that only individuals with the proper permissions can perform specific actions, reducing the risk of unauthorized access or unintended modifications.

3.3 Portability

To make the CS104 Robotics Triathlon software more portable, consider the following:

- Minimize code specific to the robot's hardware.
- Choose a programming language that works on different robots.
- Avoid code tied to a particular compiler or operating system.
- Ensure the software behaves the same across various environments.
- Reduce reliance on robot-specific hardware.
- Use modular and flexible libraries.
- Implement standard communication methods.
- Explore containerization to enhance portability.
- These steps will help the software run on various robots and systems while maintaining consistent performance.

4. Requirements Confirmation/Stakeholder sign-off

Meeting Date	Attendees (name and role)	Comments
12/4/23	Anthony/Project Manager, Peter/Systems Designer, Lostada/Software Developer	Github was submitted
12/4/23	Anthony/Project Manager, Peter/Systems Designer, Lostada/Software Developer	Github was reviewed

5. System Design

5.1 Algorithm

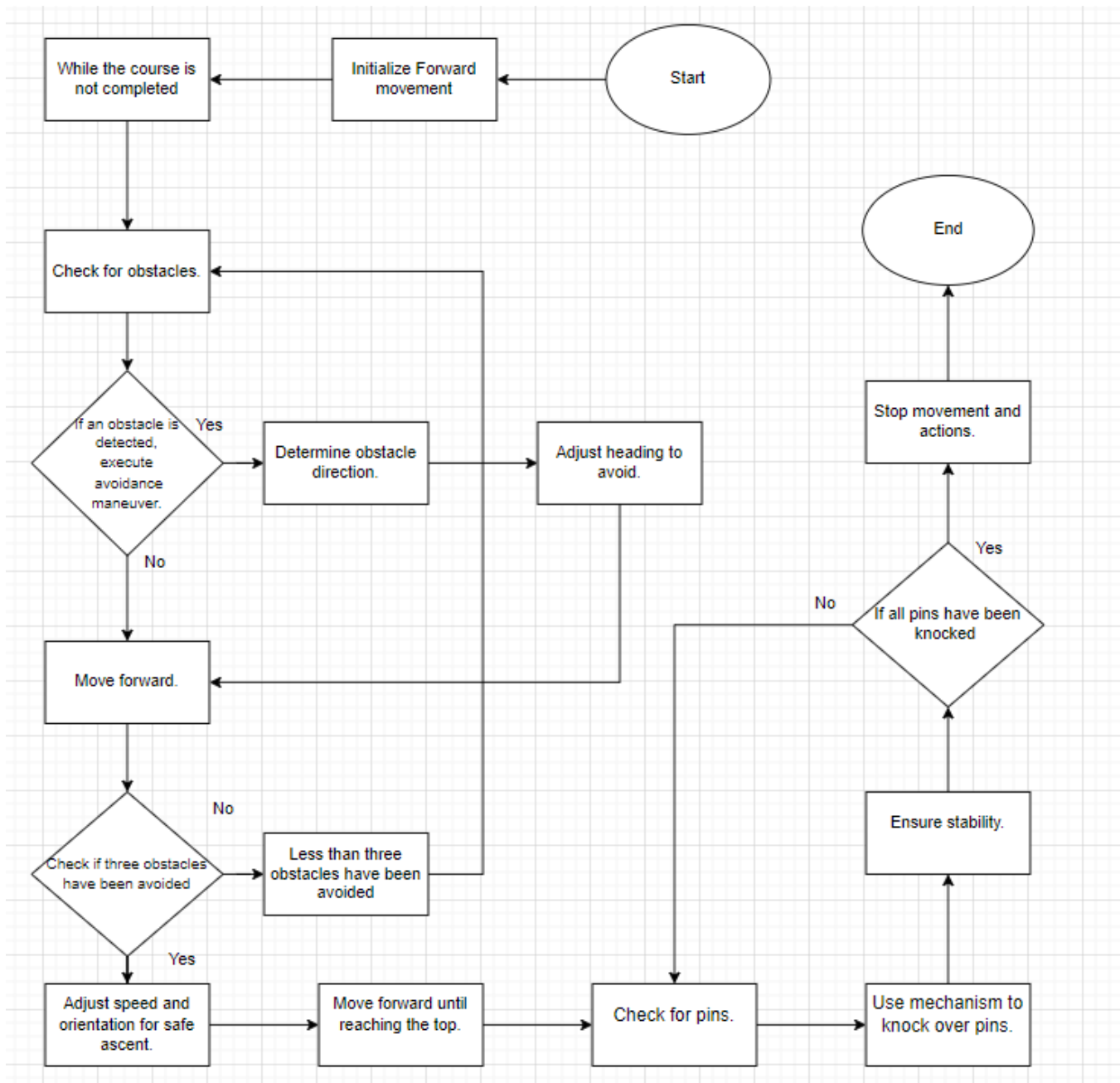
Initialization:

1. Initialization:
 - a. Begin in the center of the starting square.
2. Movement Loop:
 - a. While the course is not completed:
 - i. Check for obstacles.
 - ii. If an obstacle is detected, execute avoidance maneuver.
 - iii. If no obstacle, move forward.
3. Object Avoidance:
 - a. If obstacle detected:
 - i. Determine obstacle direction.
 - ii. Adjust heading to avoid.
 - iii. Continue moving forward.
4. Ramp Navigation:
 - a. After avoiding three obstacles:
 - i. Detect ramp incline.
 - ii. Adjust speed and orientation for safe ascent.

Sprint 3 - Agility Design Document

- iii. Move forward until reaching the top.
- 5. Pin Knocking:
 - i. Check for pins.
 - ii. Use mechanism to knock over pins.
 - iii. Ensure stability.
- 6. Course Completion:
 - a. If all steps are completed, the course is done.
- 7. End Execution:
 - a. Stop movement and actions.

5.2 System Flow



5.3 Software

We used the Sphero block coding platform.
Components of Block Code:

Blocks: In block code, each coding instruction is represented by a visual block. These blocks typically have a shape and a specific color code to indicate their purpose.

Snap Together: Blocks are designed to snap together like puzzle pieces. This snap-together feature enforces correct syntax, preventing common coding errors.

Visual Representation: Instead of writing text-based code, programmers arrange and connect blocks to create a program. The visual representation simplifies coding concepts. The code is seen below:



5.4 Hardware

In the context of the CS104 Robotics Triathlon project using the Sphero ball robot, the hardware platforms used for development, testing, and demonstration would primarily involve the Sphero robot itself and additional hardware components for testing and integration. Here's a description of the relevant hardware platforms:

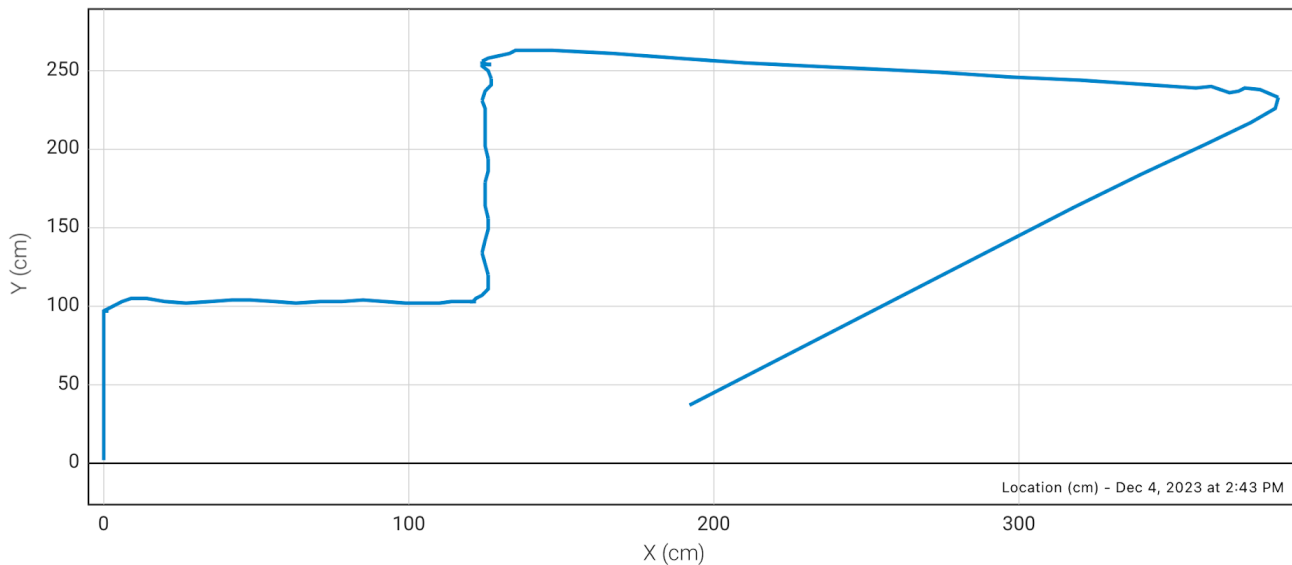
Sphero Ball Robot:

The central hardware platform for the project is the Sphero ball robot. This spherical robot is equipped with sensors, motors, and Bluetooth connectivity. It serves as the primary robot to navigate the triathlon course.

Sprint 3 - Agility Design Document

Sensor Modules:

Depending on the project's requirements, additional sensor modules may be used, such as proximity sensors, ultrasonic sensors, or cameras. These sensors can be used for obstacle detection and avoidance. The sensor data diagram is seen below:



For development and testing, computer workstations or laptops may be used to write and test code, design algorithms, and simulate robot behavior using development environments.

Charging Stations:

Charging stations are needed to ensure the Sphero robot is fully charged and ready for each triathlon event.


5.5 Test Plan

Reason for Test Case	Test Date	Expected Output	Observed Output	Staff Name	Pass/Fail
Confirm proper setup	12/4/23	Ensure the robot starts at the correct location	Verify that the robot begins its course from the designated starting point.	Anthony M	Pass
Robot must go over the ramp	12/4/23	Confirm that the robot does not go off course after going over the ramp.	The robot successfully was able to maintain it's path after going on the ramp.	Peter S.	Pass
Check if it follows the path	12/4/23	Verify the robot navigates the periphery	Confirm that the robot successfully travels along the room's perimeter without colliding with obstacles.	Anthony M	Pass

Sprint 3 - Agility Design Document

Reason for Test Case	Test Date	Expected Output	Observed Output	Staff Name	Pass/Fail
Check obstacle avoidance	12/4/23	Test the robot's ability to avoid obstacles	Assess the robot's capability to detect and avoid obstacles along its path.	Anthony M	Pass
Verify completeness of the route	12/4/23	Confirm the robot returns to the starting point	Ensure that the robot completes the course and returns to its initial position.	Anthony M	Pass
Make sure that the robot completes the square path.	12/4/23	Confirm that the robot will accurately traverse the correct path.	It was successfully able to repeat the steps as it traversed to its desired position.	Anthony M	Pass
Must Knock over pins	12/4/23	It will knock over all of the pins possible.	The robot's position was not compromised as it toppled over all of the pins.	Anthony M	Pass
Verify robot's Sprint	12/4/23	To finish the assignment	Concluded the assignment by confirming that all test cases have been executed successfully.	Anthony M	Pass

5.6 Task List/Gantt Chart

 Sprint 3 agility Gantt project plan Template.xlsx

Sprint 3 - Agility

Select a period to highlight at right. A legend describing the charting follows.

Period Highlight: ▼

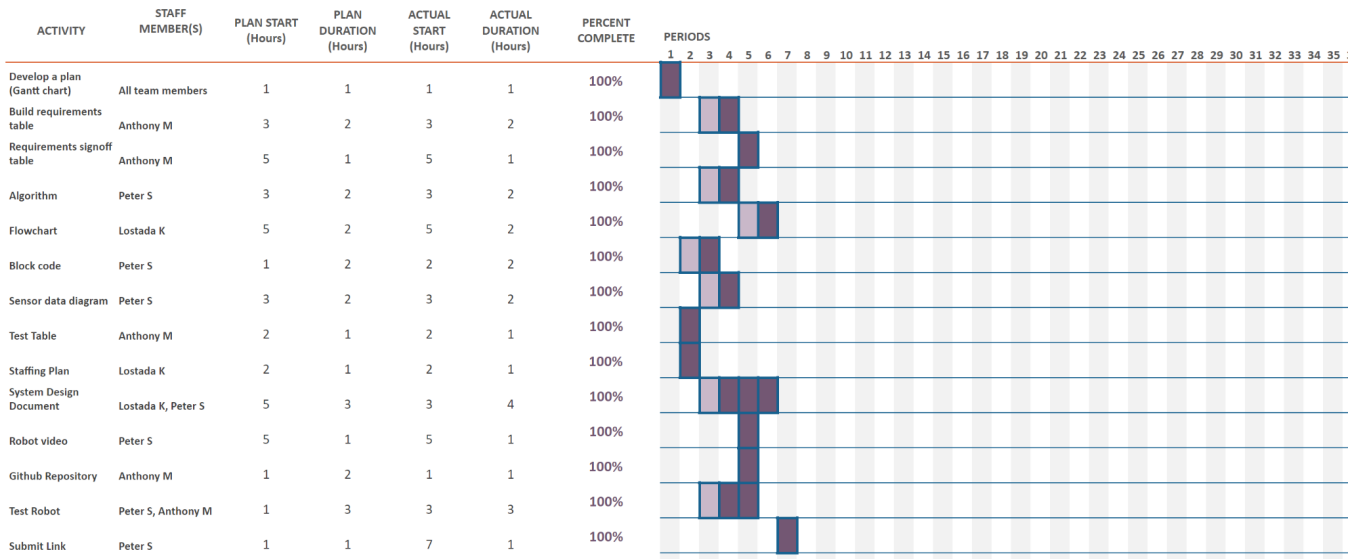
Plan Duration

Actual Start

% Complete

Actual (beyond plan)

% Complete (beyond plan)



5.7 Staffing Plan

Sprint 3 - Agility Design Document

Name	Role	Responsibility	Reports To
Develop a plan (Gantt chart)		All team members	All
Build requirements table	Business Analyst	Anthony M	All
Requirements signoff table	Project Manager	Anthony M	All
Algorithm	Software Developer	Peter Silvio	All
Flowchart	Systems Designer	Lostada K	All
Block code	Software Developer	Peter Silvio	All
Sensor data diagram	Systems Designer	Lostada K	All
Test Table	Quality Assurance (QA) Engineer	Anthony M	All
Staffing Plan	Project Manager	Lostada K	All
System Design Document	Systems Designer	Lostada K, Peter Silvio	All
Robot video	Videographer	Peter Silvio	All
Github Repository	Version Control Manager	Peter Silvio	All
Test Robot	Technician	Anthony M	All
Submit Link	Project Manager	Peter Silvio	All